REMARKS

These Remarks are in reply to the Final Office Action mailed March 18, 2008. Claims 1, 4-7, 11, 13, 17, 20-23, 27, 28, and 36-39 were pending in the Application prior to the outstanding Office Action. Claims 1 and 17 are currently being amended. No claims are currently being canceled or added. Accordingly, claims 1, 4-7, 11, 13, 17, 20-23, 27, 28, and 36-39 remain for the Examiner's consideration, with claims 1, 11, 17, and 27 being independent. Reconsideration and withdrawal of the rejections are respectfully requested.

I. Claims Rejected under 35 U.S.C. §103(a)

Claims 1, 4-7, 11, 13, 17, 20-23, 27, 28, and 36-39 were rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Cloud et al. (U.S. Patent No. 6,253,369, hereinafter Cloud) in view of Kim et al. (U.S. Publication No. 2002/0065701, hereinafter Kim).

II. Summary of Examiner Interview

Applicants thank the Examiner for the telephone interview conducted on June 17, 2008. In the interview, Applicants' undersigned representative proposed amending claim 1 to include the term "independent," and explained how the specification supports this term, and how adding this term further distinguishes claim 1 from the cited references, as discussed below. The Examiner agreed that the term "independent" was supported by the specification, and that this amendment may overcome the claim 1 rejection in view of the cited references, but the Examiner indicated he may need to perform a new prior art search.

III. Claim 1

A. Independent Tasks as Required by Claim 1

Independent claim 1 has been amended to more clearly define that the tasks in one element of the claim are independent tasks. In particular, as amended, claim 1 requires that the procedural part contains logic enabling the batch job execution system to perform execution of *independent* individual tasks separately, in parallel. Take for example a batch job having two independent tasks. Support in the specification for these independent tasks can be found at p. 17, lines 6-10, which provides the example of a batch job that may contain a set of Word files and a

set of gif files, both of which are to be converted into HTML files. Such a job may have two tasks, one would include the Word files that are to be converted to HTML, and the other would include the gif files that are to be converted to HTML. Since these tasks are *independent* of one another, they may be performed in parallel.

It was alleged in the Office Action (p. 3, para. 3) that the claim 1 element that requires execution of individual tasks separately, in parallel, is disclosed by Kim (p. 8, para. 0173), which describes parent and child processes executing separately and in parallel. However, the parent and child processes are not independent, as explained below.

Kim discloses processes that are combined under parent-child relationships by using the sub-process activity. The sub-process can be executed in an asynchronous manner, such that a parent process and a child process (sub-process activity) execute separately and in parallel. In particular, a child process starts its execution when the workflow of the parent process reaches the sub-process activity. The parent process continues the activities following the sub-process activity, without waiting for the completion of the child process. (paras. 0167, 0172, and 0173). Because the parent process calls the child process, however, the parent-child relationship is a dependent one. The particular element of claim 1 under discussion, on the other hand, requires independent tasks. Thus, the dependent parent-child relationship of processes, as disclosed in Kim, is not the same as independent individual tasks executing separately, in parallel, as required by claim 1.

Additionally, Kim does not teach or suggest independent activities performed in parallel, as discussed further here. Kim teaches a process comprising activities that are performed in a logical order according to a business rule, where the business rules are conditions for completing an activity. (p. 2, paras. 0040 and 0042). Kim discloses that a scheduler manages activities to direct workflow from activity to activity. (para. 0193). Because in Kim the workflow flows from activity to activity, any independent activities are performed serially, not in parallel. Thus, any serially performed independent activities disclosed in Kim are not the same as independent individual tasks executing separately, in parallel, as required by claim 1.

B. Additional Steps as Required by Claim 1

As amended, claim 1 further requires that the procedural part does not know about the scheduling contained in the declarative part, but can specify *additional steps* that must be

performed after the procedural part completes before a particular task is considered to have completed. Applicants were faced with the problem of how to specify steps to be reliably executed with maximum parallelism when some steps depend upon the output of other steps and when it is not knowable in advance which steps must be executed. The steps that must be executed as part of the job may depend upon the job's input data and are not knowable in advance. In order to solve this problem, these features of claim 1 require that the procedural part can specify additional steps that must be performed after the procedural part completes but before a task is completed.

Support in the specification for these additional steps can be found at p. 18, line 10 through p. 19, line 3, which states the following: The procedural part carries out the arbitrary logic of the tasks and can augment the set of steps to be performed. For example, a procedural part that converts the contents of a zip file to HTML may decompress and expand the zip file into four other files and return four steps to convert each of these files to HTML. These new steps are reported back to the job management apparatus 104 and become the next set of steps to be completed.

It was alleged in the Office Action (p. 3, para. 2) that this claim 1 element of additional steps is disclosed by Kim (p. 8, para. 0173), which states that the child process that can be executed in an asynchronous manner. For at least the following reasons, Applicants respectfully disagree.

As discussed above under "Independent Tasks as Required by Claim 1," for the parent-child process relationship where the child process can be executed in an asynchronous manner, as disclosed in Kim, the parent process *calls* the child process. In other words, in Kim, the child process begins before the parent process ends. The procedural part of claim 1, however, *returns* steps to be completed. In other words, the steps in claim 1 are performed after the procedural part completes. Thus, the parent-child process relationship where the child process can be executed in an asynchronous manner, as disclosed in Kim, is not the same as a procedural part that can specify additional steps that must be performed after the procedural part completes before a particular task is considered to have completed, as required by claim 1.

C. Claim 1 Conclusion

As such, Applicants respectfully submit that Kim fails to teach or suggest that the procedural part contains logic enabling the batch job execution system to perform execution of independent individual tasks separately, in parallel; and that the procedural part can specify additional steps that must be performed after the procedural part completes before a particular task is considered to have completed, as required by claim 1. Further, Cloud does not teach or suggest these deficiencies of Kim with regard to claim 1. For at least these reasons, Applicants respectfully submit that that the embodiment defined in claim 1 is neither anticipated by nor obvious in view of Cloud or Kim, taken alone or in combination, and respectfully request reconsideration of the claim.

IV. Claim 11

A. Task Information as Required by Claim 11

Independent claim 11 requires making a call to *put*, which transfers at least a portion of the information in the task to be executed to the remote platform; making a call to *convert*, which instructs the remote platform to perform a function on the information transferred to the remote platform; and making a call to *get*, which retrieves the converted information from the remote platform.

Each of these three "making a call" elements involves related "information," or "the information in the task to be executed." For example, making a call to put transfers a portion of this information, making a call to convert performs a function on the information, and making a call to get retrieves the converted information. Applicants respectfully submit that portions of Cloud cited in the Office Action that allegedly perform these tasks do not deal with related information, as discussed below.

It was alleged in the Office Action (p. 9, para. 2) that *making a call to put*, as required by claim 11, is disclosed by Cloud (col. 20, line 55-col. 21, line 5), which states "sending to the host." This portion of Cloud includes sending to a host data extracted from a data entry screen of a terminal. As an aside, data extracted from a data entry screen as disclosed in Cloud is not the same as a portion of the information in a task of a batch job, as required by claim 11. It was alleged in the Office Action (p. 9, para. 3) that *making a call to convert*, as required by claim 11, is disclosed by Cloud (col. 18, lines 50-58). This portion of Cloud states that files are referenced

during the batch work flow object generation process in which the parameters are converted into CICS command level source code, then compiled and linked into an executable module. As disclosed in Cloud, however, the information allegedly involved in making a call to put (data entry screen information) is completely different than the information allegedly involved in making a call to convert (file parameters converted to source code). Claim 11, on the other hand, requires that the information transferred in the step of making a call to put is the same information that is converted in the step of making a call to convert.

It was alleged in the Office Action (p. 9, para. 4) that *making a call to get*, as required by claim 11, is disclosed by Cloud (col. 11, lines 29-42). In this portion of Cloud, processing which begins with a request and ends with a reply is called a unit of work. To complete a complex unit of work, the work flow will decompose the message received and invoke several tasks to independently retrieve information from whatever different sources are necessary. As disclosed in Cloud, however, the information retrieved in making a call to get (from whatever different sources) is completely different than the information in making a call to convert (file parameters converted to source code). Claim 11, on the other hand, requires that the converted information from the step of making a call to convert is the same information that is retrieved in the step of making a call to get.

B. Repeating Steps as Required by Claim 11

Claim 11 further requires *repeating* each step of making a call to put, convert, and get until the task is completed. It was alleged in the Office Action (p. 9, para. 5) that this element of claim 11 is disclosed by Cloud (col. 14, lines 25-49). In this portion of Cloud, units of work objects A and B can be executed concurrently and that unit of work C must await the completion of units of work A and B for the work flow to complete. Work objects that must complete for the work flow to complete, however, is different than repeating steps, as required by claim 11. Further, Cloud does not appear to teach or suggest repeating of any steps. Thus, Cloud does not teach repeating each step of making a call to put, convert, and get until the task is completed, as required by claim 11.

C. Claim 11 Conclusion

As such, Applicants respectfully submit that Cloud fails to teach or suggest making a call to put, which transfers at least a portion of the information in the task to be executed to the remote platform; making a call to convert, which instructs the remote platform to perform a function on the information transferred to the remote platform; making a call to get, which retrieves the converted information from the remote platform; and repeating each step of making a call to put, convert and get until the task is completed, as required by claim 11. Further, Kim does not teach or suggest these deficiencies of Cloud with regard to claim 11. For at least these reasons, Applicants respectfully submit that that the embodiment defined in claim 11 is neither anticipated by nor obvious in view of Cloud or Kim, taken alone or in combination, and respectfully request reconsideration of the claim.

V. Claim 17

Independent claim 17 has been amended in a similar manner as claim 1 to more clearly define the embodiment therein. As amended, claim 17 is directed to an apparatus that includes a client that performs similar features to those discussed above with reference to claim 1. For similar reasons to those discussed above with respect to claim 1, Applicants respectfully assert that Kim fails to teach or suggest that that the procedural part contains logic enabling the batch job execution system to perform execution of independent individual tasks separately, in parallel; and that the procedural part can specify additional steps that must be performed after the procedural part completes before a particular task is considered to have completed, as required by claim 17. Thus, Applicants respectfully submit that the embodiment defined by claim 17 is likewise neither anticipated by, nor obvious in view of Cloud or Kim, taken alone or in combination, and respectfully request reconsideration of the claim.

VI. Claim 27

Independent claim 27 is directed to an apparatus that includes a service provider that performs similar features to those discussed above with reference to claim 11. For similar reasons as provided above with respect to claim 11, Applicants respectfully assert that Cloud fails to teach or suggest making a call to put, which transfers at least a portion of the information in the task to be executed to the remote platform; making a call to convert, which instructs the

remote platform to perform a function on the information transferred to the remote platform; making a call to get, which retrieves the converted information from the remote platform; and repeating each step of making a call to put, convert and get until the task is completed, as required by claim 27. Thus, Applicants respectfully submit that the embodiment defined by claim 27 is likewise neither anticipated by, nor obvious in view of Cloud and Kim, taken alone or in combination, and respectfully request reconsideration of the claim.

VII. Dependent Claims

Claims 4-7, 13, 20-23, 28, and 36-39 are not addressed separately, but it is respectfully submitted that these claims are allowable for at least the reason that these claims depend from allowable claims discussed above. It is also submitted that each of these claims also add their own limitation which render them patentable in their own right. Applicants respectfully reserve the right to argue these limitations should it become necessary in the future.

VIII. Conclusion

In light of the above, it is respectfully requested that all of the claims now pending in the subject patent application should be allowable, and a Notice of Allowance is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if she can assist in any way in expediting issuance of a patent.

No fee is believed due in connection with this paper. However, the Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: June 18, 2008

By: /Julie Daniels Missud/
Julie Daniels Missud
Reg. No. 51,330

FLIESLER MEYER LLP
650 California Street, 14th Floor
San Francisco, CA 94108
Telephone: (415) 362-3800
Customer Number 23910